SYMBOLICS

S-RENDER

GRAPHICS

DIVISION

symbolics

AYC5

S-Render

August 1985

S-Render

August 1985

This document corresponds to System Release 6.0, S-Render Version 10.

This document was prepared by the Graphics Division of Symbolics, Inc.

The software, data, and information contained herein are proprietary to, and comprise valuable trade secrets of, Symbolics, Inc. They are given in confidence by Symbolics pursuant to a written license agreement, and may be used, copied, transmitted, and stored only in accordance with the terms of such license.

This document may not be reproduced in whole or in part without the prior written consent of Symbolics, Inc.

Copyright © 1985, Symbolics, Inc. All rights reserved.

Symbolics, Symbolics 3600, Symbolics 3670, Symbolics 3640, SYMBOLICS-LISP, ZETALISP, MACSYMA, Document Examiner, S-DYNAMICS, S-GEOMETRY, S-PAINT, and S-RENDER are trademarks of Symbolics, Inc.
PDP-11, UNIBUS, and VAX are trademarks of Digital Equipment Corporation.
MULTIBUS is a trademark of Intel Corporation.

Restricted rights legend.
Use, duplication, or disclosure by the government is subject to restrictions as set forth in subdivision (b)(3)(ii) of the Rights in Technical Data and Computer Software Clause at FAR 52.227-7013.

#998026

Printed in the USA.

CONTENTS

GETTING STARTED 1

Entering S-Render 1

Using the Mouse 2

BASIC CONCEPTS 3

Rendering Models 3

Opacity 5

Normals 5

Interpolation 6

Color by Vertex 7

Opacity by Vertex 7

Light Sources 8

USING THE ATTRIBUTES MENU 9

Specifying Colors 12

Specifying Light Sources 12

Specifying Light Databases 16

EXERCISE 18

S-RENDER

S-Render is a software package that is used in conjunction with S-Paint, S-Geometry, and S-Dynamics to produce visually realistic 3D objects and scenes. S-Render creates still images, or, when used in conjunction with S-Dynamics, creates a series of images that form an animated sequence. S-Render accepts polygonal models constructed in S-Geometry and renders them taking into account the:

- . light sources
- . surface characteristics
- . positions of the surfaces and sources
- . orientations of the surfaces and sources

You should be familiar with basic S-Geometry and/or S-Dynamics operations before you use S-Render. See the documents *S-Geometry* and *S-Dynamics.*

You can assign attributes to the polygonal model to specify the color, opacity, shading model, and light sources to use during the rendering process. Attributes can be assigned to an entire polyhedron, or can be specified per polygon.

Note: While it is possible to initially load S-Geometry without S-Render, this is not recommended if S-Render is to be used later in the session. When objects with rendering attributes assigned are read in under these conditions, they might not be converted into the current format that S-Render uses. The conversion is done only during the reading of objects from files, and the particulars of the conversion process can vary with the version of S-Render. Therefore if S-Render is subsequently loaded after objects have been read into S-Geometry or scripts loaded into S-Dynamics, the proper conversion might not take place. This can result in S-Render complaining about unknown polygon flavor types when it is passed the unconverted objects. Once the converted objects have been written out, they are updated permanently.

GETTING STARTED

Entering S-Render

From S-Geometry:

[Render] in the Output menu of S-Geometry.

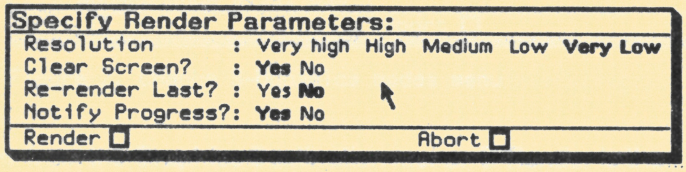

Specify Render Parameters:
Resolution : Very high High Medium Low Very Low
Clear Screen? : Yes No
Re-render Last? : Yes No
Notify Progress?: Yes No
Render
Abort

Figure 1. Specify Render Parameters menu

If you enter S-Render from S-Geometry, the Specify Render Parameters menu appears. The menu items are:

[Resolution]

Allows you to specify a magnification factor to control the resolution at which S-Render creates an image.

[Clear Screen?]

Specifies whether the old image in the Render window is cleared before the new image is written.

[Re-render Last?]

Specifies whether the previously rendered scene should be rerendered, under perhaps changed lighting conditions. Several steps of the rendering process are skipped, thereby decreasing the computation time. This option assumes that you did not change any models or attributes of the models since the previous invocation.

[Notify Progress?]

Specifies whether comments are shown in the 3D Live Window. These comments indicate the progress of S-Render as its various sequential phases are initiated.

From S-Dynamics:

[Change Modes] at the bottom right corner of the screen. A Change S-Dynamics modes menu appears.

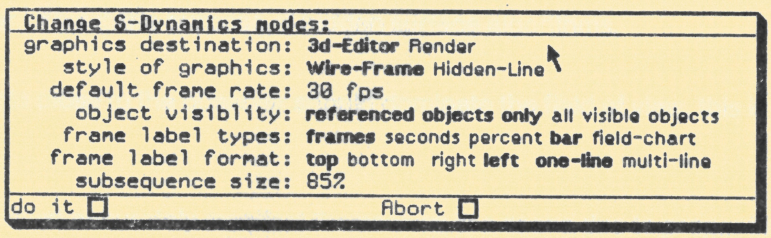

Change S-Dynamics modes:
graphics destination: 3d-Editor Render
style of graphics: Wire-Frame Hidden-Line
default frame rate: 30 fps
object visiblity: referenced objects only all visible objects
frame label types: frames seconds percent bar field-chart
frame label format: top bottom right left one-line multi-line
subsequence size: 85%
do it
Abort

Figure 2. Change S-Dynamics modes menu

Select [Render] from the Graphics Destination menu. S-Render is then invoked automatically when S-Dynamics is subsequently used to animate a scene or display a single frame. All visible objects and a camera view are passed from S-Geometry to S-Render, which constructs a shaded image in the Render Window.

Using the Mouse

The mouse is a pointing device that you move around on a flat surface. These motions are sensed by the Lisp Machine, which responds by moving a cursor around on the screen in a corresponding manner. The shape of the cursor varies, depending on context.

The three buttons on the mouse are referred to as left, middle, and right. They are used to specify operations to be performed. S-Render's menus are mouse-sensitive; that is, clicking one of the mouse buttons on a menu item causes some action to occur. When you move the mouse cursor over a menu item, an outline box appears around the item. Clicking on the boxed item causes the action to occur.

Mouse actions are represented in the following manner:

Appearance	Representing
[Color]	Menu item, always click left.
(R), (M), (L)	Mouse clicks: R-click right, M-click middle, L-click left.

FUNCTION X moves the mouse between the black-and-white screen and the color screen.

BASIC CONCEPTS

To render a visually realistic scene, the following rules need to be observed.

First, objects that are opaque must obscure objects that lie behind them, from the point of view of the observer. This is the purpose of hidden surface algorithms.

Second, objects close to the observer should dominate the field of view, this is known as perspective.

Third, objects must accurately manifest themselves in a scene due to proper interaction between their material properties and the properties of illuminating bodies. This is the purpose of shading models.

Rendering Models

Shading models are used to simulate physical (and sometimes nonphysical) aspects of material surfaces, as they interact with light sources. S-Render rendering models combine a shading model and those aspects of the rendering algorithm that affect the application of the shading model, such as interpolation. The method of interpolation between the sampled application of the shading model on a polygonal surface can strongly affect the perception of what sort of material is being simulated.

S-Render uses two shading models: *Lambert* and *Phong*.

Lambert shading is based on Lambert's Law, which states that the intensity of light emitted from a material is independent of direction (for a perfectly diffusive material). Lambert's Cosine Law adds the notion that intensity per unit surface area proportionately decreases as the surface normal becomes increasingly inclined with respect to the light source. See the section "Normals."

The proportionality factor is the cosine of the angle between the two vectors. If the angle is zero, then the light is fully reflected. If the angle is greater than or equal to 90 degrees, no light is reflected.

The effect of solely employing Lambert shading is that the surface appears to be matte. This can result in a chalky appearance to the surface, or appear like a waxy object internally lit. This surface property is called *diffusive*.

Phong shading accounts for the ability of materials to have some degree of mirror-like reflectivity, or specularity, as well as being diffusive. The *specular highlight* is computed to be in a direction where the angle of reflectance matches the angle of incidence. The geometry of the model asserts that the light vector, the normal vector, and the highlight vector all lie in the same plane, as in figure 3. The intensity of light reflected in the highlight direction is proportional to how closely the eye vector and the highlight vector line up.

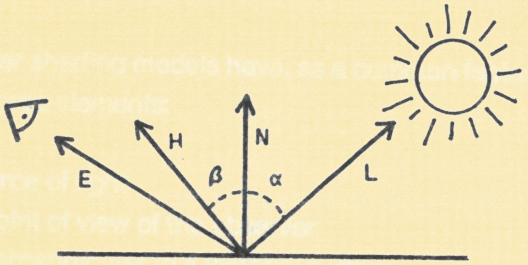


Figure 3. Phong shading specular highlight

The proportionality factor is related to the cosine of angle between the two sources. However, the cosine factor is modified by raising it to a power. The higher the power, the shinier the surface appears, since the effect is to concentrate the specular highlight (from a point light source) into an increasingly smaller disk.

An ambient component is also used in Phong shading that accounts for light generated internally by the object, and emitted regardless of the orientation of the surface normal vector or the eye vector.

The ambient, diffusive, and specular components of Phong shading are all individually specifiable as colors. For example, if white light illuminates a surface that is simulating red plastic, the ambient component is black, the diffusive component red, and the specular component white. A metallic material, on the other hand, has an off-white specular component, and a glowing object has a nonblack ambient component.

All light sources interact with the diffusive (or surface color) component of either Lambert or Phong shading models; however, only point and line light sources interact with the specular component of Phong shading. Light sources do not interact with the ambient component of Phong shading. The effects of an ambient light source and the ambient component of Phong shading are independent.

Opacity

Opacity, like surface color, is another factor in light-object interactions. A fully opaque object does not permit any of the scene behind it to show. Partially opaque to fully transparent objects pass a proportional amount of the background scene information to the front of the object. In the current model such objects are totally transparent internally, that is, the surface color of transparent objects does not filter the light transmitted through it. In addition, refraction is not modelled, and lens type effects do not occur.

Normals

All S-Render shading models have, as a common factor, a geometric relationship between the following three elements:

. a source of light
. the point of view of the observer
. the perpendicular to a material surface

The perpendicular to a material surface is called the "surface normal," "normal to the surface," or just plain "normal." This is usually represented diagramatically as an arrow pointing away from the surface of an object, as in figure 4. The shaft of the arrow forms a 90 degree angle to the surface of the object. Mathematically it is called a vector. It is calculated for a polygonal surface by employing the cross product (or vector product) operator of vector algebra, or for a smoothly curved surface by employing the gradient operator of vector calculus. For a mathematically specified surface, the computation of the normal to the surface (at any given point on the surface) is a straightforward operation.

The light source and observer point of view directions are also represented as vectors, as in figure 5. They are easily computed by taking the difference between the position of the light source (eye) and the position of the material surface where the lighting model is to be applied. The vector is then normalized, that is, the distance property of the vector is divided out, leaving only directional information.

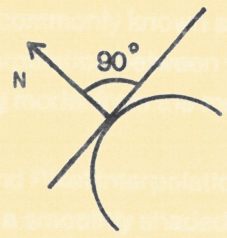


Figure 4. Surface normal

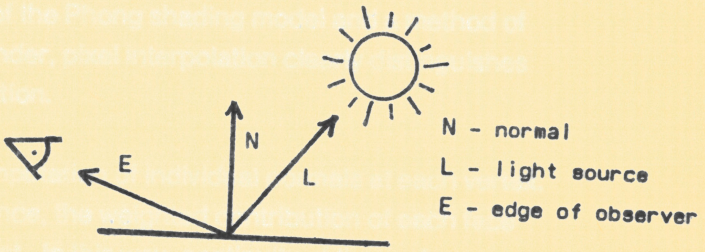


Figure 5. Light source and observer point of view represented as vectors

Interpolation

A polygonal rendering model description is not complete without specifying a method of interpolation, because the method of interpolation can strongly affect the perception of what sort of material is being simulated. S-Render supports three methods of polygonal interpolation: *faceted*, *Gouraud*, and *pixel*.

Faceted is the simplest form of interpolation. Faceted interpolation chooses a representative point within a polygonal face, for example the center. An averaged normal is computed from the edges, and is assumed to be constant across the face. The shading model is applied to the position and normal, and a color is computed, which is used for the entire face. The surface of a polyhedral object then appears to be constructed of hard-edged facets. The light sources appear to be infinitely far away, since displacement across the face produces no change in lighting value. The faces do catch the light sources, but since highlights are not possible within the face, the effect is like that of a polished gem in a dull environment.

Gouraud interpolation applies the shading model to each vertex of the polygonal face. The resulting colors at the vertices are linearly interpolated along the edges across the face of the polygon. Since the values obtained for linear interpolation cannot exceed the maximum or minimum values at the endpoints, highlights tend to radiate out from the vertices and edges. Under animation they also tend to flick from one edge to another, since a highlight maxima cannot smoothly traverse across a face under this form of interpolation. In addition, artifacts called *mach bands* cause an object to show pronounced ridges in an otherwise smooth surface, due to discontinuities in surface curvature from this method of interpolation.

Pixel interpolation applies the shading model to every pixel within a polygonal face. The normals computed at the vertices are interpolated along the edges, and then across the face of the polygon. These interpolated normals are fed to the shading model along with the position of the pixel. This produces an acceptably smooth surface. In addition, light sources that are not placed at infinity are accurately modelled as forming a narrowing locus of light as they approach a polygonal face.

Pixel interpolation allows you to simulate Phong shading highlights within a face. Phong shading is commonly known as a combination of the Phong shading model and a method of normal interpolation between vertices. In S-Render, pixel interpolation clearly distinguishes the shading model from the method of interpolation.

Gouraud and Pixel interpolation require the computation of individual normals at each vertex. To achieve a smoothly shaded surface appearance, the weighted contribution of each face sharing the vertex needs to be taken into account. In this way, continuity of the surface normals are maintained across edges.

S-Render supports two ways to weight the contributions of the faces to derive the vertex normals: *face, and vertex.*

Face is really a throwback to faceted interpolation, using a constant normal everywhere within the polygon. Using face as the vertex normal source results in the appearance of a flat surface. However, using Gouraud interpolation can still result in some shading variations across the face, since the position of the vertices with respect to local light sources influence the geometry of the lighting model. Pixel interpolation produces variations as well, since the position of each pixel, like that of the vertices, is distinct. In addition, the Phong shading model is sensitive to eye position with respect to the surface, and can generate realistic shading variations in the highlights, even if local lights are not in use.

Vertex weights the normal derived from each face's cross product computation by the subtended angle between the edges. (See figure 6.)

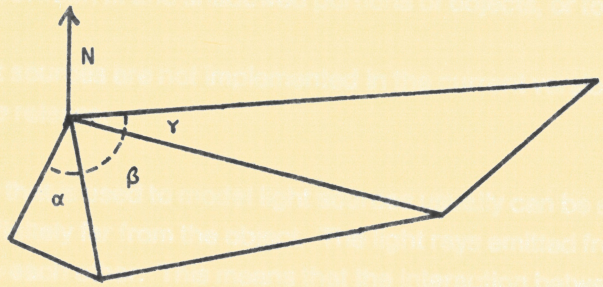


Figure 6. Angle weighting to derive the vertex normal

Color by Vertex

A Gouraud type linear interpolation is used in the Color by Vertex option to Surface Color. The surface color of a face is individually specified at each vertex, and then interpolated across the face of the polygon. With Gouraud shading interpolation, the lighting model is applied only at the vertices - the surface color specified at the vertex is taken into account by the shading model, and the resulting vertex color is interpolated across the polygon. With Pixel shading interpolation, the interpolated surface color is factored into the shading model at each pixel.

Opacity by Vertex

A Gouraud type linear interpolation is also used by the Opacity by Vertex option to Opacity. The opacity is specified at each vertex, and then interpolated across the face of the polygon. This option is independent of the shading model or type of shading interpolation.

Light Sources

The light sources in S-Render, unlike physical light sources, do not model the property that light intensity incident on a surface is inversely proportional to the square of the distance between light source and object. Nor is the intensity of light emitted/reflected from an object to the eye modelled as a function of the distance between the two. Although this is nonphysical, it is convenient for many scene modelling applications.

Three light source types are implemented in S-Render: point, ambient, and line.

Point light sources are the most common. Point is a dimensionless point in space that emits light. Stars are examples of point light sources that are placed at infinity.

Ambient light sources are directionless. They model light that has been bounced off a number of objects, so that it seems to be present everywhere in space. The effect is to reduce contrast between lit and shadowed portions of objects, or to blur small shape details.

Line light sources are not implemented in the current version of Render, but will be supported in a future release.

The math that is used to model light sources usually can be simplified if the light sources are placed infinitely far from the object. The light rays emitted from such an object are then parallel to each other. This means that the interaction between an object and such a light source is dependent only on the orientation of the object with respect to the light source, not the position of the object. This has modelling advantages as well, since a translation of an object does not change the pattern of light falling on it, if the observer is placed far enough away from the object.

For example, if a point light source is placed a finite distance from an object (that is, a local light source), as the object is moved closer to the light source, the pattern of illumination shrinks to a smaller and smaller radius, until finally it is extinguished when the object envelopes the source. If the light source is placed at infinity, only a rotation of the object with respect to the light source changes how the object is illuminated.

The color of light reflected from a material surface is modified by the interaction. S-Render simply multiplies the red, green, and blue components of the light source color with the surface color of the object. For example, white light incident on another white surface reflects as white light. White light incident on a red surface reflects as red light. Blue light incident on a red surface is not reflected at all. Yellow light (a mixture of red and green light) reflects from a red surface as red light. If the surface color is less than full value, the light it reflects is proportionately dimmer.

USING THE ATTRIBUTES MENU

The Attributes menu sets the attributes of polyhedra, faces (polygons), and vertices. The Attributes menu for polyhedron and face type objects are identical. Faces that are not individually specified inherit their attributes from the parent polyhedron.

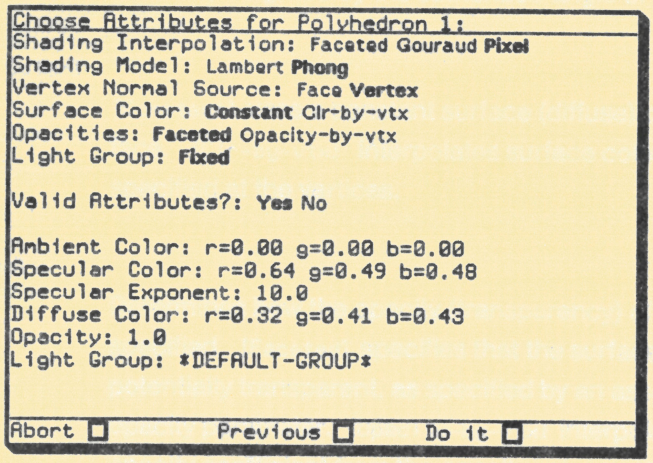


Figure 7. Attributes menu

The Attributes menu is divided into three sections. The first section specifies attributes.

[Shading Interpolation]

Specifies whether the shading model is applied once per face, once per vertex, or for every pixel. [Faceted] means the color for the face is the same for every pixel within the face. [Gouraud] corresponds to Gouraud interpolation, meaning it applies the shading model once per vertex, and interpolates the color across the face. [Pixel] invokes the shading model for every pixel within the face, and interpolates the normals from the vertices (Phong interpolation).

[Shading Model]

Specifies which of two lighting models to use wherever the shading model is applied. [Lambert] corresponds to a simple cosine law falloff with respect to surface normal and each light source. [Phong] corresponds to a more complex model, which adds a background ambient color to be added into the light reflected by the surface, as well as a specular reflectance component for each light source. The falloff of the specular component can be controlled via an exponent. The color that the surface presents to a specular component is specifiable, and in both Lambert and Phong models the color of the surface for the cosine law falloff is also specifiable.

[Vertex Normal Source]

Determines how the surface normal is specified at the vertices. [Face] forces the use of a fixed face normal for each vertex of the face. [Vertex] specifies that vertex normals are determined from an average of all shared face normals, weighted by the subtended angle of the face.

[Surface Color]

[Constant] uses a constant surface (diffuse) color for the face. [Clr-by-vtx] interpolates surface color from colors specified at the vertices.

[Opacities]

Determines how the opacity (transparency) of a surface is specified. [Faceted] specifies that the surface is potentially transparent, as specified by an associated opacity parameter. [Opacity-by-vtx] interpolates the opacity coefficient from the opacity values established at the vertices, in the same way that [Clr-by-vtx] interpolates surface color.

[Light Group]

Has a single option, [Fixed], that specifies the light group for shading.

The second section of the Attributes menu consists of a single entry:

[Valid Attributes?]

A few combinations of attributes selections in the first section are invalid or meaningless. When you specify an invalid combination, [No] is highlighted. S-Render ignores attempts to set invalid attributes for an object when leaving the Attributes menu. [Yes] is highlighted when the combination is meaningful.

The third section of the Attributes menu contains entries that correspond to the parameters that need to specified, based on the attributes selected in the first section.

[Diffuse Color]

Specifies colors. See the section "Specifying Colors."

[Light Group]

Changes the group of light sources that impinge on a polyhedron or face, or invokes the light database editor. (You can also click (R) on [Render] in the [Output] menu of S-Geometry to invoke the light database editor.) Click (L) to display a menu of defined light groups. Click (M) to invoke the light database editor. See the section "Specifying Light Sources."

The Attributes menu for vertices has one fixed section. This section contains parameter settings for attributes that interpolate parameters from the vertices, such as [Clr-by-vtx] and [Opacity-by-vtx]. Vertex attribute parameters are specified independently of the polyhedral or face attributes. If the latter's attributes call for such parameters, they are used, otherwise they are ignored.

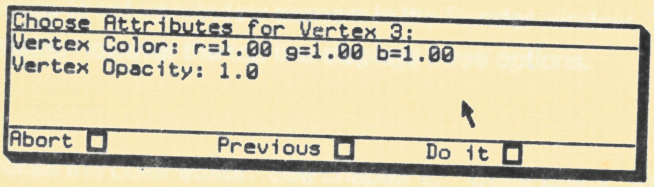


Figure 8. Vertex attributes menu

Three options appear at the bottom of both Attributes menus.

[Abort]

Leaves the attributes of the selected object unchanged from the previously stored values, and exits the menu.

[Do it]

Stores the currently specified values into the object, and exits the menu.

[Previous]

Sets the attribute specifications to match the attribute settings corresponding to the most recent use of [Do it]. The menu is not exited, allowing further customization of the values before [Do it] is used to store them on the object, or [Abort] is used to abandon the effort. [Previous] is handy to set the attributes for a number of objects to the same or slightly modified values.

Specifying Colors

You can specify the color parameters present in the attribute menus and the light source parameter specification menus in the light database editor in two ways.

(M) displays a menu that allows you to type in color values for red, green, and blue. Typically these values should be between 0.0 and 1.0.

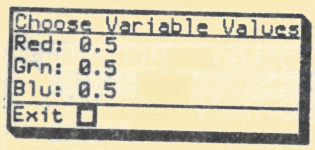


Figure 9. Color values menu

(L) exposes a color palette in the Render window. If this is the first invocation of the palette, use the mouse to set up window corners in the Render window. This palette is similar to the S-Paint palette, however the top line displays three options.

[edit]

Invokes the color editor. See chapter 9, "Color Editor" in the document *Color System*.

[type]

Displays a window that is identical to the one that would have exposed if (M) had originally been used on the color entry, except that this one exposes on the Render window.

[add]

Adds a palette entry corresponding to the original definition of the color. This is useful to add a color that was specified elsewhere into the palette system.

Rather than choosing one of these three options, you can click on a color bar on the palette to get a previously defined color. Any color defined by [edit], [type], or [add] is automatically added to the complement of colors displayed on the palette.

Specifying Light Sources

Light sources are specified through the light database editor. To enter the light database editor, use [Light Group (M)] in the Attributes menus for Polyhedra and Faces, or use [Render (R)] in the Output menu in S-Geometry.

When the light database editor is initialized, a default source named *DEFAULT-LIGHT* is created, and placed into a default group, named *DEFAULT-GROUP*. Light groups drive the shading models for Polyhedra and Faces. All objects, unless specified otherwise in the Light Group attribute parameter, look to the *DEFAULT-GROUP* for their light sources.

When you enter the light database editor, the first menu you see is the Dispatcher menu. You can access the rest of the light database functions from this menu. The first column specifies an operation. The second column specifies an object.

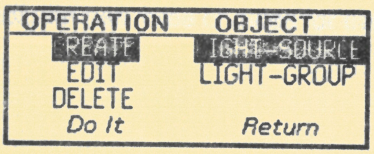


Figure 10. Dispatcher menu

The menu exit options are:

[Do It]

Invokes the currently selected function.

[Return]

Exits the light database editor, and saves all changes.

The menu choice options are:

[CREATE]

The light sources and groups are created by first specifying names in a menu that pops up when [Do It] is used in the dispatcher. The names can be completely arbitrary since they are, as a class, generated as symbols within their own individual packages. For example, you can name a light source and light groups the same name, and the names will not collide with symbols specified elsewhere in any system package. Since light groups carry no attributes other than their list of light sources, their creation is finished after a name has been chosen.

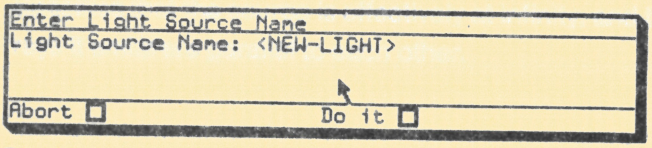


Figure 11. Light source name menu

However, Light Sources need to be further defined. The next specification menu for them consists of three columns.

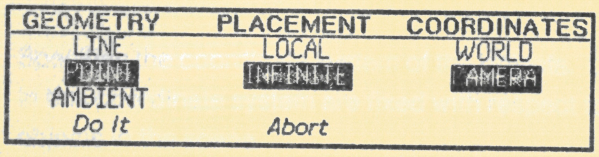


Figure 12. Light source attributes specification menu

The first column has three choices for the geometry of the light source: [Line], [Point], and [Ambient].

[Line]

Specifies a light source that is an infinitely long line in space, that is, a one dimensional source.

[Point]

Specifies a zero dimensional point source.

[Ambient]

Provides sourceless light evenly distributed throughout space, interacting with the diffuse color component of any lighting model. This differs from the ambient component of the Phong shading model, which contributes its color independently of the diffuse color component of a surface, or any light source. Since an ambient light source is isotropic with respect to location and orientation, the other two columns of this menu do not pertain to it.

The second column determines placement of the source.

[Local]

Specifies that the source is in near space, and that the light rays it emits diverge toward illuminated surfaces.

[Infinite]

Specifies that the source is effectively at infinity, and that the light rays it emits are parallel to each other.

The third column determines the coordinate system in which the placement information is specified.

[World]

Specifies the coordinate system of the objects. Light sources specified in this coordinate system are fixed with respect to any static polyhedral objects in the scene.

[Camera]

Specifies the coordinate system relative to the camera or eye coordinate system. Light sources specified in this coordinate system are tied to the camera, and camera moves drag these sources around in order to maintain the same relationship. Put another way, the light sources behave as if they are attached to rigid poles connected to the camera.

The last light source creation menu is particularized based on the type of light source chosen. This menu specifies parameters of a light source.

All light sources have an associated intensity parameter. This parameter is a color, and can be specified in the same manner available to the color attributes of objects. This is the only parameter for an ambient light. (It makes no sense to specify more than one ambient light source per light group, since they simply add, regardless of geometry.)

Local sources have a position parameter, which corresponds to a cartesian coordinate in the selected coordinate system. World coordinates are specified in a right-handed coordinate system (increasing Z values come toward the observer), while camera coordinates are specified in a left-handed coordinate system (increasing Z values head away from the observer). Infinite sources use a direction specification, which points toward their location at infinity. This direction is a vector whose origin lies on the origin of the selected coordinate system, and passes through the point specified by the direction parameter. After specification, this coordinate point is normalized to lie on the unit sphere.

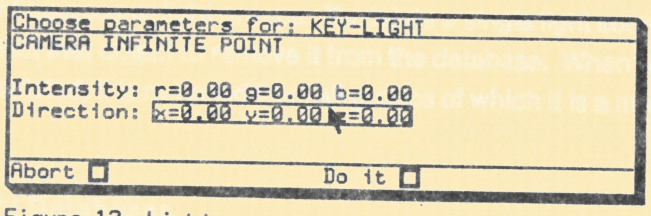


Figure 13. Light source parameter menu

Warning: If you specify light sources and object color attributes that exceed the intensity bounds of the color frame buffer, the software scales down such colors so that the red, green, and blue values do not exceed 100 percent (1.0). However, this manipulation is evident in the rendered image. The effect appears as a loss of detail in the affected portions of the image, mach bands, or photographic *bloom*. There is no explicit warning message that oversaturated pixels are being generated. In future releases, features will be added to determine what sources and model attribute parameters are contributing to such effects in a rendered image.

Returning back to the Dispatcher menu, the other items are:

[EDIT]

When you edit a light source, a menu appears that allows you to select one of the previously created light sources. Then you see a menu that is identical to the last specification menu in the light source [CREATE] sequence. The parameters values in this menu can be modified and the changes stored as the definition of the selected light source.

When you edit a light group, a menu that displays a matrix of choice boxes appears. The left column lists the names of all defined light sources. The top row contains headings of all defined light groups. Select a box to include the light source whose name appears in that row in the light group whose name appears in that column. Light sources can be members of a number of light groups.

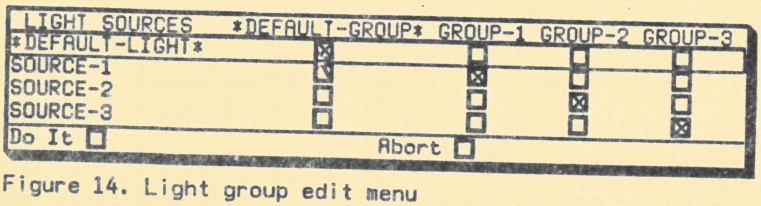


Figure 14. Light group edit menu

[DELETE]

A menu appears that displays all defined objects respectively, either light sources or light groups. To delete a single light source or group, select that object to remove it from the database. When a light source is deleted, it is removed from all groups of which it is a member.

Saving Light Databases

When a camera view is written out to a file in S-Geometry, the light database is saved along with it. When the view file is read back in, or selected, a notification menu allows you to overwrite your existing light database. Move the mouse cursor off the notification menu to inhibit the loading of the light database stored with the view. Click on the notification menu to overwrite your existing light database with the new one. See the document *S-Geometry* for more information on loading and storing camera views.

Warning: The aspect ratio of the rendered image is the same as the aspect ratio of the S-Geometry window. This is specified in the `[Camera Format]` entry of the Modify Camera Parameters menu, reached by using the `[Parameters]` entry in the S-Geometry Camera menu. Depending on the value of `[Resolution]` in the Specify Render Parameters menu, the actual aspect ratio can vary slightly from the specified value, due to limitations of the zoom-pan hardware in the color frame buffer. To get a notification when such a difference occurs, set the `[Notify Progress]` entry in the Specify Render Parameters menu to `[Yes]`, and look for a requested vs. actual aspect ratio message in the 3D Live Window.

Warning: Concave polygons, or nonplanar polygons viewed edge-on might not render correctly. If a face seems to disappear, convert it into a set of convex planar polygons by adding edges to the model. See the `[modify element]` operation "cut" for faces in S-Geometry.

Exercise

The following is a simple walk-through exercise.

In S-Geometry:

1. (R) in the main Geometry window. — Brings up the main menu.

2. [Add New Object] — Brings up the new object menu.

3. [cube] — Creates a cube.

4. [polyhedra] — Selects polyhedra mode.

5. Move the mouse over cube and (M).

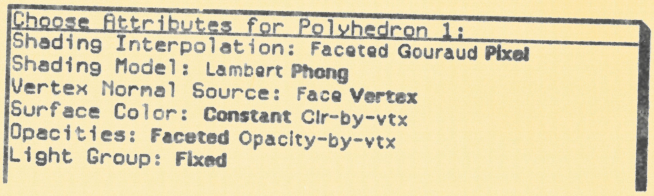
6. (R) in the main Geometry window. — Brings up the main menu.

7. [Attributes] — Brings up attributes menu.

8. [Pixel] — Selects pixel interpolation.

9. [Phong] — Selects Phong shading.

10. [Vertex] — Selects angle-weighted vertex normals.


Choose Attributes for Polyhedron 1:
Shading Interpolation: Faceted Gouraud Pixel
Shading Model: Lambert Phong
Vertex Normal Source: Face Vertex
Surface Color: Constant Clr-by-vtx
Opacities: Faceted Opacity-by-vtx
Light Group: Fixed

Choose Attributes menu with selections set.

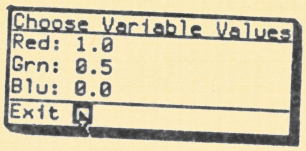
11. (M) over the diffuse color parameter list. — Brings up color type-in menu.

12. (L) over red. Type in 1.0 and press RETURN.

These three steps specify an orange color.

13. (L) over green. Type in .5 and press RETURN.

14. (L) over blue. Type in 0.0 and press RETURN.



Color values menu set.

15. [exit]

Exits Choose Variable Values menu.

16. [Do it]

Exits Attributes menu.

17. [Render] in the output menu.

Invokes S-Render.

18. [Render] in the Specify Render Parameters menu.

If a color screen has not been specified, either by means of color:demo or explicitly, a menu comes up asking for the correct buffer type. If so:

19. Select the screen configuration relevant to your installation. This will typically be 24-bit high resolution or 24-bit NTSC.

A color screen and a Render window are created, and an orange cube appears on your color monitor screen.